

Bureau of Long Term Services and Supports

Utah Electronic Visit Verification (UEVV)

Configuring Secure Messaging

Version 1.1



March 2020

Contact

Please send all inquiries and questions to dmhf_evv@utah.gov.

Table of Contents

1. Overview	1
2. Generate the client-side keystore and certificate	1
2.1. Creating the keystore	1
2.2. Export the certificate	2
3. Get the server-side certificate	3

1. Overview

Some SOAP connections are secured against public access through the use of X.509 certificates. By configuring and exchanging these certificates, SOAP messages are able to pass through such security.

The following steps outline how to configure a client-side certificate for this purpose:

1. Generate the client keystore
2. Export the client certificate and provide it to BLTSS at dmhf_evv@utah.gov.
3. Get the server certificate via the WSDL URL

The following are some common acronyms that are used throughout this document:

Acronym	Meaning
BLTSS	Bureau of Long-Term Services and Supports
EVV	Electronic Visit Verification
SOAP	Simple Object Access Protocol – An XML document which is used to transmit data
URL	Universal Resource Locator – An address to a piece of information, such as a website, document, or application endpoint.
WSDL	Web Service Description Language – An XML document which describes how to connect and make requests to a web service

2. Generate the client-side keystore and certificate

A client-side keystore needs to be generated, if one does not already exist. This is used to hold client-side encryption keys to handle the setup of a secure connection to the EVV application. A public certificate is produced from this keystore, which is provided to the Bureau of Long-Term Services and Supports (BLTSS), where it is loaded onto the server in order to allow the server to recognize a valid connection attempt.

2.1. Creating the keystore

To create such a keystore, use the “keytool” application that comes with Java. This application can be found in the `bin` subfolder of the Java directory.

For example:

```
C:\Program Files\Java\jdk1.8.0_171\bin\keytool.exe
```

Once the keytool application has been identified, open a command prompt (or PowerShell console) at that location, and execute the following command:

```
keytool -genkey -keyalg RSA -alias <<alias>> -keystore <<keystore filename  
with path>> -storepass <<password>> -validity 3600
```

This creates a new keystore file and generates a new key. Replace the following fields in that command with the correct information:

- **<<alias>>** : a name for the particular key. For example, `evvclient`
- **<<keystore filename with path>>** : a name and location for the keystore itself. Where the keystore is created, and what it will be called. For example, `c:\Users\jdoe\Desktop\myfirstkeystore.jks`
- **<<password>>** : a password to access the contents of the keystore. This should be limited to alphanumeric characters, and the special characters `!@#$$%^&*+=_-` with a total length of no more than 30 characters, to avoid possible problems with Oracle software.

Note: Password management software such as [KeePass](#) makes storing complicated passwords easier, and can handle things such as tracking password expiration dates automatically.

Upon executing the command, a prompt will appear asking a series of questions. These are tied to the new key that is being generated. The table below outlines the particular questions. Responses will be dependent upon the organization involved.

Table 1: Keystore identity information

Question
What is your first and last name?
What is the name of your organizational unit?
What is the name of your organization?
What is the name of your City or Locality?
What is the name of your State or Province?
What is the two-letter country code for this unit?
Is the provided information correct?

Finally, a password prompt will appear that corresponds to the chosen name. It can be left blank to default to the keystore password, instead.

Note: When setting a password for the key, *both* the keystore password and key password will be needed to access the new key.

2.2. Export the certificate

With the client-side keystore created, a public certificate must be exported to BLTSS. Using the same Command Prompt (or PowerShell console) from section 2.1, enter the following command to export the certificate:

```
keytool -export -rfc -keystore <<keystore filename with path>> -alias  
<<alias>> -file <<certificate filename with path>>
```

Replace the following fields in that command with the correct information:

- <<keystore filename with path>> : This is the path and filename of the keystore created in section 2.1. Use the same information, unless the keystore was moved or renamed between steps.
- <<alias>> : This is the name of the particular key that you want the certificate to be created from. Use the same alias chosen from section 2.1, when the keystore was created.
- <<certificate filename with path>> : This is the path and filename of the certificate being created. For example, c:\Users\jdoe\Desktop\evvclient.cer

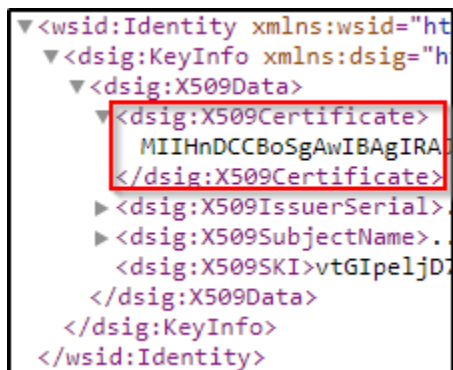
Finally, send this newly created certificate file to BLTSS at dmhf_evv@utah.gov, so that they can import the certificate into the server-side keystore.

3. Get the server-side certificate

The WSDL used to generate the SOAP UI project (see section **Error! Reference source not found.**) contains certificate information about the server-side of the connection.

The WSDL URL is provided by BLTSS. Navigating to the URL opens a page in your default web browser. Near the bottom of the WSDL is a segment titled “dsig:X509Certificate”, which contains a base-64 encoded string. This string is the necessary server-side certificate.

Figure 1: The X509 certificate is available in the WSDL



Copy the string (only the base-64 encoded text itself, not the surrounding XML tags) into an empty text document. Surround the certificate with the following lines:

```
-----BEGIN CERTIFICATE-----  
<<base64 cert from the WSDL>>  
-----END CERTIFICATE-----
```

Note: The lines in question are “BEGIN CERTIFICATE” or “END CERTIFICATE” in all caps, with 5 leading and trailing hyphens.

Save this text file. This needs to be incorporated into the system. The mechanisms for doing so will be dependent on the particular system configuration.